International Conference on Applied Internet and Information Technologies, 2016
DOI:10.20544/AIIT2016.19

Reducing Competitive Cache Misses in Modern Processor Architectures

Milcho Prisagjanec[1] and Pece Mitrevski[1]

[1] Faculty of information and communication technologies, Partizanska bb, 7000 Bitola, Macedonia
{milco.prisaganec, pece.mitrevski}@fikt.edu.mk

Abstract. The trends in the development of the multicore processors very often have the first level of cache memory implemented in. The increasing number of threads inside the cores and access competitively to the shared cache memory becomes the reason for the increased number of the competitive cache misses and decline the performances. Develop of the modern processor architectures leads to an increased number of cache misses. This paper has been made an attempt to implement the technique of decreasing the number of the competitive cache misses in the first level of the cache memory. This technique enables a competitive access to the complete cache memory when there is a hit inside of it. But if there are cache misses, then the memory data through the techniques of replacement is put in a virtual part given to the threads so that the competitive cache misses can be avoided. The results gained by using a simulator of the processor show decrease of the number of the cache misses and increase the performances for 15%. The conclusion out of this research can be that the cache misses are a real challenge for the future designers of processors and they need to be paid more attention to.

Keywords: processor, memory hierarchy, cache memory, cache misses, multicore processors, multithread processor, competitive cache misses, modern processor architectures

1. Introduction

The basic target develops the processor is increasing of the performances. The bottlenecks in the processor which decrease its performances are being eliminated by implementation of the different types of techniques in the architecture itself. If we view this historically we can notice that the increasing of the frequency, implement the out of order execution of the instruction, enlargement of the instruction window, and the parallelism of the instructional levels all these contributed to increasing the performances of the processors in some periods of time.

But, according to some authors, the gap between the processor speed and its memory which has still existed since appear the computers, despite the many offered techniques, while some of them already implemented in the commercial processors, is the reason for the decrease of the performances [2]. The instruction that accesses memory until the complete with memory data blocks the processor resources at the same time and decrees the performances. The basic technique which is being implemented all the time is the memory hierarchy. When the processor needs a memory data it access to the first level of the cache memory. If the memory data are in it, we can say that there is a hit, and the

processor loads them inside. But if not, we say that there is a cache miss and the memory system starts a procedure of elimination which can take several hundred tact cycles [3].

The researches made by many authors showed that the cache misses reduce the performances of the processors even up to 20%. Even more significant information is that the gap between the speed and the memory of the processors is getting deeper and bigger through the time, which contributes to the cache misses number. That is why there have been numerous efforts made to design techniques to reduce the cache misses. For example, the prefetching technique [5], which has been implemented into the commercial processors, supplies taking over the memory data and fetching them into the cache memory before the processor gets to them. Some other authors [6], [7], in their works have given ideas of a virtual enlargement of the instruction window, so that it would be possible to unlock the processor's resources in the window and withdrawing in advance the memory data which are to be used by the processor in the near future.

In the modern processor architectures the trends of implementation of bigger number of cores and threads in the processor itself and the shared memory between them, are the reason for increase the number of the cache misses and the decrees performances. Among them appear competitive cash failures by a common access memory system. There are even some techniques that are used for reduction of the cache misses, for example the prefetching techniques, contribute now to the appearance of such competitive cache misses and decrees performances.

Taking that into consideration, we, as authors of this work, propose a technique for reduction of the competitive cache misses. The idea is, every thread to have its own virtual cache memory in the first level in the process of replacement of the blocks, while during the process of loading, all the cache memory to be used. The proposed technique to the simulator of a processor showed great increasing of the performances of the processor up to 15%. As positive sides of the technique that we propose are the easy way of implementation, and the fact it doesn't change revolutionized the present architecture of the processor. But as a negative point can be noticed that the cache memory is poorly used in the first level in the process of not very intensive workload of the processor.

The researches of our work certified again that the cache misses significantly reduce the number of the performances of the processor. The reduction of its number is a challenge to the future explorers and processor designers.

2. Competitive Cache Misses and Performance Impact

The inability to break through the technological reductions in the processor design, changes the direction of their development. That means that the idea is to design an architecture which would perform multiple numbers of instructions at the same clock frequency. The new concept is based on the previously known system in the supercomputers where more computers parallel execute their workload. This architecture is modified by using more processor cores to be placed on one silicon crystal. Each of the cores is a processor for itself, but the new architecture significantly reduces dissipation of the energy. The trends of increasing the number of cores and the threads in them are more present in the modern architectures of processors. The new technology increases the pipeline of the instruction through the processor, but that means necessity of a larger instructional window. By doing so, there is increasing of the number of the instructions

which have greater need of the memory data from the memory system, whose latency is too big and causes stagnation in the work of the processor.

2.1. Performances of Multi Core Processors

In an ideal case, the double core processor should accomplish a programmer code in a double speed, compared to the single core processor. But the practical cases so far showed that it doesn't really happen, and that the double core processor is only 1.5 times faster than the single core processor. In certain cases, the single core processors even showed better performances. One of the reasons is the cache misses.

In [8] there has been done an analysis of the cache misses depending on the number of the cores. By using a simulator there has been designed a multicore processor. This processor has got only L1 cache memory with capacity of 512Kbytes, the instruction cache memory of 256Kbytes and the data cache memory of 256Kbytes. The bus supports MESI protocol for maintenance of the coherency of the memory. The capacity of the main memory is 1 GB. By using a simulator is executing a benchmark of the designed processor. All the time the testing process is being made by changing the number of cores in the processor. The results shown can be read from the figure 1.

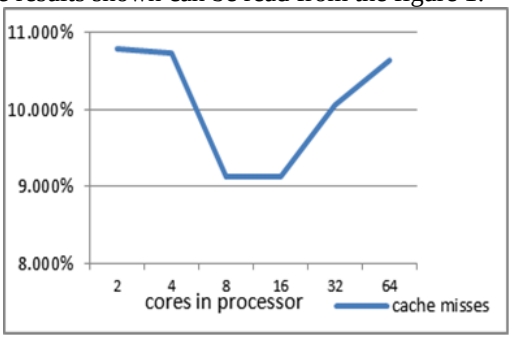

Fig. 1. How the number of processor cores affects cache misses

From the figure we can see that the number of the cache misses for the simulated architecture of the processor is 10% of the number of the memory access. But it can also be concluded from the figure that by increasing the number of cores from 16 to more, the number of cache misses also increases which has a negative effect to the performances of the processor.

The idea in the modern multi core architecture is to improve the pipeline of the instructions by increasing the number of cores and the threads in each of them. That would provide parallel execution of a greater number of instructions that means enlargement of the instruction window. It would hide the latency of the memory which brings to increasing the number of the performances. But the increasing of the number of the cores and threads leads to increasing of the number of the cache misses, causing decrease the processor performances. One of the reasons for that is the competitive cache misses.

2.2. Competitive Cache Misses

In modern processor architectures, more cores and threads share the same memory. The workload that the processor has to accomplish is divided into processes. The processes are independent entities, which can be paralleled accomplished in the threads. In the modern processor architecture, more threads use shared cache memory. Every process needs certain memory data, and if it cannot provide them from the first level of the memory, it loads them from the memory system by using a certain method for change of the blocks. This method of change does not check if the block, which is being changed, would be used in the near future by some other process. It would bring to removal of some memory blocks, which would be needed by some other active processes in the future cycles. It would cause a decrease processor performance.

This scenario would get even worse if we use the prefetching technique to load the memory data from the first level of the cache memory. If every process would use this technique to load the memory data from the first level of cache memory, which it would need in the near future, due to the restricted capacity of the cache memory and its shared use, would cause competitive memory access and mutual removal of the blocks of data for the processes. It would cause the appearance of the cache misses. The possibility of appearance of the competitive cache memory access grows with processor workload, then, with the increasing of the number of the cores and the threads which share the same memory, and finally because of the restricted capacity of the cache memory of the processor.

3. Reducing Competitive Cache Misses – Techniques and Benefits

To avoid the competitive access of the threads, we suggest this technique of a one-way shared cache memory. The cache memory in the first level is shared to as many parts as the number of the threads. All threads competitively access the memory. If a thread contains an instruction, which accesses the memory, check that there is the required memory data. Therefore, if this data can be found in the first level of the cache memory, then it is being loaded. It is the same as in the modern processor architecture. If the data can not be found in the first level of the cache memory, then it is necessary to be loaded by the upper level of the memory system. The loading of the memory block is completed by using one of the modern techniques for replacing, but by doing this the block can be loaded only in the part of the cache memory which belongs to the thread. That means that during the loading of the memory data this thread uses all the shared cache memory from the first level. But if a certain data doesn't exist in the first level of the memory, then it loads it from the memory system, but in the her virtual part of the cache memory.

The prefetching technique, which is frequently used in the modern architecture of the processors, is also a reason for the appearance of the competitive cache misses. This technique is implemented in the architecture of the processor in order to decrease the latency of the memory and the number of the cache misses. This means that for the instructions which have the access to the memory this technique makes early loading of the memory data and places them in the first level of the cache memory. At the moment of execution of the instruction the memory data are already in the cache memory and the processor just takes them from there with minimal delay.

But in the modern processor architectures, there is competitive access of the threads to the first level of the cache memory with the prefetching technique. That is the reason for the competitive access to the same memory address while loading the memory data from the upper levels of the cache memory. That is the reason why it is necessary in the offered technique the prefetching technique to load the memory blocks to the second level of the cache memory.

3.1. Performance Benefits

In order to get measurable values of the performances of the processor architecture, which could be compared, we designed a processor simulator with programming language python. By using this simulator, we tested the performances of an ideal processor which doesn't have cache misses, a multithread processor and a processor where we implemented the offered technique for non-competitive access to the cache memory in the first level. The results gained from the testing are shown in figure 2 where you can see the time of execution of the benchmark by using different simulated architectures of the processor. The offered technique is implemented in the multi thread processor architecture which is with one core with 4 threads. The simulator makes parallel execution of the instructions in all of the threads. When the simulator comes to an instruction with access to the memory, it checks whether it can be found in the L1 cache memory. If there is a a hit, it loads the memory data and completing instruction in the processor. But if there is a miss in L1 cache memory, the simulator searches this memory data in the upper levels of the memory. The simulator uses the technique of block replacement and loads the memory data only in the virtual part of L1 cache memory which belongs to the matching thread.

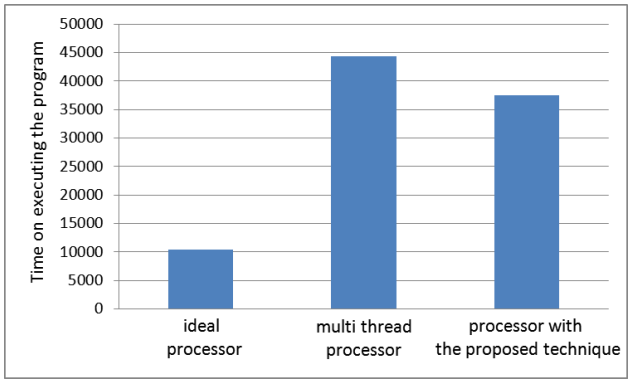

Fig. 2. Graphical overview of the performances of three types of processors

According to the results shown the offered technique makes the time of execution of the programme faster for 15%. The reason is the decreased number of the competitive cache misses to the first level of the cache memory.

The good side of the offered technique is that it doesn't revolutionary change the architecture of the processors. It can be implemented very easily. The change comes only in the technique of replacing the blocks in the first level of the cache memory. In the

cache memory are implemented a tag which is used for marking which block belongs to which thread.

While replacing the memory block we use this tag for virtual division of the cache memory of the number of the threads which access to the memory.

The prefetching technique is not something new, this technique is used in modern processors. The change means that this technique early load memory data in the second level of the cache memory, to avoid the competitive access to the first level of the cache memory. This technique doesn't increase the processor complexity, and so, it doesn't cause the additional dissipation of the energy.

The disadvantages of this technique are the low exploitation of the first level of the cache memory during the poor workload on the processor. In cases where we have a small number of processes that reduced the number of active threads, reduced the capacity of the cache memory in the first level.

4. Conclusions

The results of this paper showed again the huge effect of the cache misses on the performances of the processor. In order to hide the latency of the memory, which is more expressed in the modern processors, there is memory hierarchy implemented in the processor architecture. But it also causes cache misses and decreasing processor performances.

This paper and the results of the simulation showed that the decreased processor performances due to the cache misses could not be ignored. The offered technique succeeds in decreasing the number of the cache misses for 15%, and it increases the processor performances.

The results of the researches show that the number of the cache misses in the modern architectures is still very big. On the other side, the direction in which they develop, by increasing the number of cores and threads and using shared cache memory, brings about the appearance of bigger number of cache misses even more, and decreasing processor performances. That is why the decreasing of the number of the cache misses is a great challenge to the future explorers and designers.

References

1. Tullsen, Dean M. and Jeffery A. Brown: Handling long-latency loads in a simultaneous multithreading processor in Proceeding of 34th annual ACM/IEEE international symposium on Microarchitecture, IEEE Computer Society, pp120-122, (2001)
2. Carvalho, C.: The gap between processor and memory speeds In Proc. of IEEE International Conference on Control and Automation (2002).
3. Hennessy, J. L., & Patterson, D. A.: Computer architecture: a quantitative approach. Elsevier (2012).
4. Smith, A. J. Cache memories. ACM Computing Surveys (CSUR), 14(3), 473-530, (1982)
5. Kim, D., Liao, S. S. W., Wang, P. H., Cuvillo, J. D., Tian, X., Zou, X., ... & Shen, J. P. :Physical experimentation with prefetching helper threads on Intel's hyper-threaded processors. In Proceedings of the international symposium on Code generation and

optimization: feedback-directed and runtime optimization (p. 27). IEEE Computer Society (2004, March).
6. Mutlu, O., Stark, J., Wilkerson, C., & Patt, Y. N.: Runahead execution: An alternative to very large instruction windows for out-of-order processors. In High-Performance Computer Architecture In Proceedings The Ninth International Symposium on (pp. 129-140). IEEE, (2003, February).
7. Zhou, H., & Conte, T. M.: Enhancing memory-level parallelism via recovery-free value prediction. Computers, IEEE Transactions on, 54(7), 897-912, (2005).
8. Prisaganec, M., Mitrevski, P.: Cache misses challenge to modern processor architectures, Proc. of the XLVIII International Scientific Conference on Information, Communication and Energy Systems and Technologies (ICEST 2013), Vol. 1, pp. 273-276, Ohrid, Macedonia (2013).